

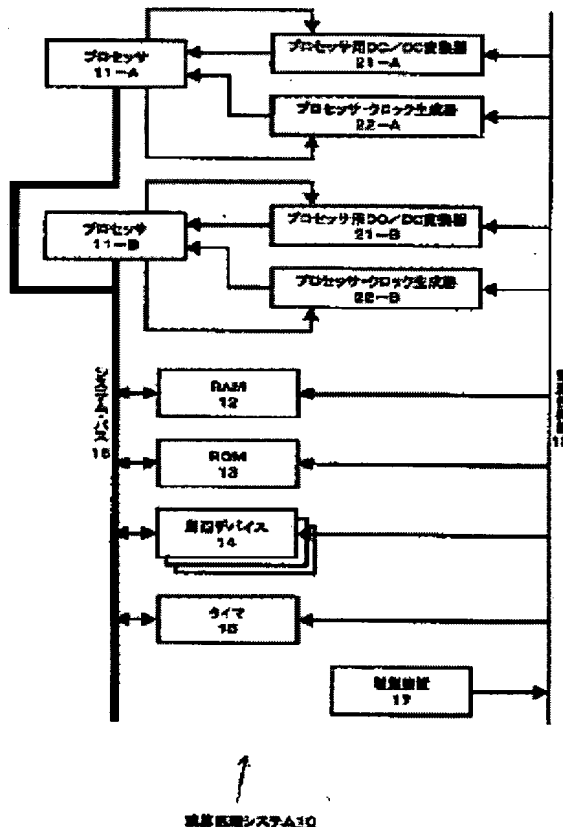
SYSTEM OF COMPUTING PROCESSING, CONTROL METHOD SYSTEM FOR TASK CONTROL, METHOD THEREFOR AND RECORD MEDIUM

Patent number: JP2002099433
 Publication date: 2002-04-05
 Inventor: TOGAWA ATSUSHI
 Applicant: SONY CORP
 Classification:
 - international: G06F1/04; G06F1/32; G06F9/46; G06F1/04; G06F1/32; G06F9/46; (IPC1-7): G06F9/46; G06F1/04; G06F1/32
 - european:
 Application number: JP20000287883 20000922
 Priority number(s): JP20000287883 20000922

Report a data error here

Abstract of JP2002099433

PROBLEM TO BE SOLVED: To provide a configuration system of multi-processor capable of reducing power consumption of a processor along with responding to real-time demands from applications. **SOLUTION:** In the configuration system having a plurality of processors enabling to change dynamically an operating frequency and a power source voltage by controlling of an operating system, the power consumption of each processor and a whole of the system is reduced with determining the most suitable power source voltage responding to a time-varying operating frequency in conjunction with adaptable changing of the frequency necessary to perform a started periodical real-time task and a non real-time task without delay.



Data supplied from the esp@cenet database - Worldwide

THIS PAGE BLANK (USPTO)

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2002-99433

(P2002-99433A)

(43) 公開日 平成14年4月5日 (2002.4.5)

(51) Int.Cl. ⁷	識別記号	F I	テーマコード* (参考)
G 0 6 F 9/46	3 4 0	G 0 6 F 9/46	3 4 0 B 5 B 0 1 1
			3 4 0 E 5 B 0 7 9
1/32		1/04	3 0 1 C 5 B 0 9 8
1/04	3 0 1	1/00	3 3 2 Z

審査請求 未請求 請求項の数16 O L (全 20 頁)

(21) 出願番号 特願2000-287883(P2000-287883)

(22) 出願日 平成12年9月22日 (2000.9.22)

(71) 出願人 000002185

ソニー株式会社

東京都品川区北品川6丁目7番35号

(72) 発明者 戸川 敦之

東京都品川区北品川6丁目7番35号 ソニー株式会社内

(74) 代理人 100101801

弁理士 山田 英治 (外2名)

Fターム(参考) 5B011 LL02 LL13

5B079 BA01 BB01 BC01 DD20

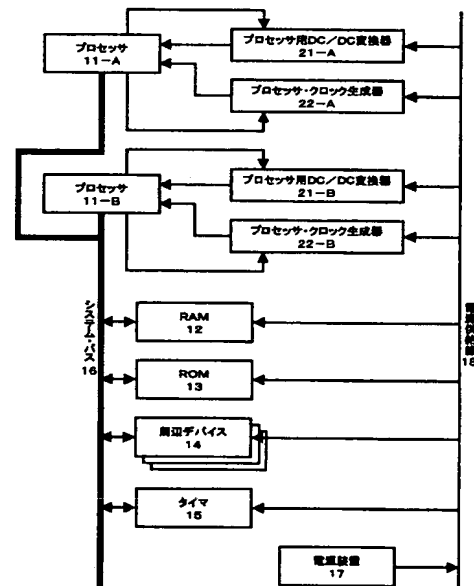
5B098 FF03 FF08 GA02 GA04

(54) 【発明の名称】 演算処理システム及び演算処理制御方法、タスク管理システム及びタスク管理方法、並びに記憶媒体

(57) 【要約】

【課題】 アプリケーションのリアルタイム要求に応えつつプロセッサによる電力消費を削減することができるマルチプロセッサ構成システムを提供する。

【解決手段】 動作周波数と電源電圧をオペレーティング・システムの制御により動的に変化させることができるプロセッサを複数備えたマルチプロセッサ構成システムにおいて、各プロセッサ毎に、起動された各タスクを遅滞なく処理するために必要な動作周波数を適応的に変化させるとともに、時々刻々と切り替わる動作周波数に応じて最適な電源電圧を決定していくことで、各プロセッサ並びにシステム全体の消費電力を低減する。



演算処理システム10

【特許請求の範囲】

【請求項 1】 次の周期の開始までに実行を完了させる必要がある周期リアルタイム・タスクと実行完了時刻の制約がない非リアルタイム・タスクを並行して実行可能なプロセッサを複数個含んだ演算処理システムであって、各プロセッサ毎に、稼動中の各時点において、各周期リアルタイム・タスクに課せられた要求を満たすために十分な動作周波数を算出する動作周波数算出手段と、各プロセッサ毎に、前記動作周波数算出手段による算出結果に基づく動作周波数クロックを該プロセッサに供給するプロセッサ・クロック生成手段と、各プロセッサ毎に、前記動作周波数算定手段により算出された動作周波数で該プロセッサを駆動させるために十分な電源電圧を算出する電源電圧算出手段と、各プロセッサ毎に、前記電源電圧算出手段による算出結果に基づく電源電圧を該プロセッサに供給するプロセッサ電源供給手段と、を具備することを特徴とする演算処理システム。

【請求項 2】 各プロセッサの稼動中の各時点において、与えられたタスクのうちいずれを実行すべきかを選択して実行するタスク選択・実行手段をさらに備えることを特徴とする請求項 1 に記載の演算処理システム。

【請求項 3】 各プロセッサの稼動中の各時点において、実行すべきタスクがなくなったことに応答して、該プロセッサを稼動率が低下したスリープ状態に移行させるスリープ遷移手段をさらに備えることを特徴とする請求項 1 に記載の演算処理システム。

【請求項 4】 各周期リアルタイム・タスクに課せられた、次の周期の開始時間、周期、一周あたり処理負荷などの情報を、タスク実行前に記録するタスク情報記録手段と、前記タスク情報記録手段により記録された情報を参照して、電源電圧と動作周波数が前記プロセッサの限界を越えず、及び／又は、既登録の周期リアルタイム・タスクに課せられた要求を満たされるように、新たな周期リアルタイム・タスクの登録を制限するタスク登録制御手段と、を各プロセッサ毎にさらに備えることを特徴とする請求項 1 に記載の演算処理システム。

【請求項 5】 各プロセッサ毎に、非リアルタイム・タスクが存在しないときに前記動作周波数算出手段によって算出された動作周波数及び／又は前記電源電圧算出手段によって算出された電源電圧が所定の下限値を下回ったことに応答して、該プロセッサの動作周波数及び／又は電源電圧を該下限値に設定する下限値設定手段をさらに備えることを特徴とする請求項 1 に記載の演算処理システム。

【請求項 6】 次の周期の開始までに実行を完了させる必要がある周期リアルタイム・タスクと実行完了時刻の制約がない非リアルタイム・タスクを並行して実行可能な複数個のプロセッサによる演算処理を制御する演算処理

制御方法であって、

各プロセッサ毎に、稼動中の各時点において、各周期リアルタイム・タスクに課せられた要求を満たすために十分な動作周波数を算出する動作周波数算出ステップと、各プロセッサ毎に、前記動作周波数算出ステップによる算出結果に基づく動作周波数クロックを該プロセッサに供給するプロセッサ・クロック生成ステップと、各プロセッサ毎に、前記動作周波数算定ステップにより算出された動作周波数で該プロセッサを駆動させるために十分な電源電圧を算出する電源電圧算出ステップと、各プロセッサ毎に、前記電源電圧算出ステップによる算出結果に基づく電源電圧を該プロセッサに供給するプロセッサ電源供給ステップと、を具備することを特徴とする演算処理制御方法。

【請求項 7】 各プロセッサの稼動中の各時点において、与えられたタスクのうちいずれを実行すべきかを選択して実行するタスク選択・実行ステップをさらに備えることを特徴とする請求項 6 に記載の演算処理制御方法。

【請求項 8】 各プロセッサの稼動中の各時点において、実行すべきタスクがなくなったことに応答して、該プロセッサを稼動率が低下したスリープ状態に移行させるスリープ遷移ステップをさらに備えることを特徴とする請求項 6 に記載の演算処理制御方法。

【請求項 9】 各周期リアルタイム・タスクに課せられた、次の周期の開始時間、周期、一周あたり処理負荷などの情報を、タスク実行前に記録するタスク情報記録ステップと、前記タスク情報記録ステップにより記録された情報を参照して、電源電圧と動作周波数が前記プロセッサの限界を越えず、及び／又は、既登録の周期リアルタイム・タスクに課せられた要求を満たされるように、新たな周期リアルタイム・タスクの登録を制限するタスク登録制御ステップと、を各プロセッサ毎にさらに備えることを特徴とする請求項 6 に記載の演算処理制御方法。

【請求項 10】 各プロセッサ毎に、非リアルタイム・タスクが存在しないときに前記動作周波数算出ステップによって算出された動作周波数及び／又は前記電源電圧算出ステップによって算出された電源電圧が所定の下限値を下回ったことに応答して、前記プロセッサの動作周波数及び／又は電源電圧を該下限値に設定する下限値設定ステップをさらに備えることを特徴とする請求項 6 に記載の演算処理制御方法。

【請求項 11】 次の周期の開始までに実行を完了させる必要がある周期リアルタイム・タスクと実行完了時刻の制約がない非リアルタイム・タスクを並行して実行可能な複数個のプロセッサのタスクを管理するタスク管理システムであって、

各プロセッサ毎に、各周期リアルタイム・タスクに課せられた、次の周期の開始時間、周期、一周あたり処理負荷などの情報を、タスク実行前に記録するタスク情

報記録手段と、
各プロセッサ毎に、該プロセッサに課された処理負荷に応じて該プロセッサの電源電圧と動作周波数を設定する動作設定手段と、

各プロセッサ毎に、前記タスク情報記録手段により記録された情報を参照して、電源電圧と動作周波数が該プロセッサの限界を越えず、及び／又は、既登録の周期リアルタイム・タスクに課せられた要求が満たされるように、新たな周期リアルタイム・タスクの登録を制限するタスク登録制御手段と、を具備することを特徴とするタスク管理システム。

【請求項 1 2】非リアルタイム・タスクが存在しないときに前記動作周波数算出手段によって算出された動作周波数が所定の下限値を下回ったことに応答して、プロセッサの動作周波数及び／又は電源電圧を該下限値に設定する下限値設定手段を各プロセッサ毎にさらに備えることを特徴とする請求項 1 1 に記載のタスク管理システム。

【請求項 1 3】次の周期の開始までに実行を完了させる必要がある周期リアルタイム・タスクと実行完了時刻の制約がない非リアルタイム・タスクを並行して実行可能な複数のプロセッサのタスクを管理するタスク管理方法であって、

各プロセッサ毎に、各周期リアルタイム・タスクに課せられた、次の周期の開始時間、周期、一周期あたりの処理負荷などの情報を、タスク実行前に記録するタスク情報記録ステップと、

各プロセッサ毎に、該プロセッサに課された処理負荷に応じて該プロセッサの電源電圧と動作周波数を設定する動作設定ステップと、

各プロセッサ毎に、前記タスク情報記録ステップにより記録された情報を参照して、電源電圧と動作周波数が該プロセッサの限界を越えず、及び／又は、既登録の周期リアルタイム・タスクに課せられた要求が満たされるように、新たな周期リアルタイム・タスクの登録を制限するタスク登録制御ステップと、を具備することを特徴とするタスク管理方法。

【請求項 1 4】非リアルタイム・タスクが存在しないときに前記動作周波数算出ステップによって算出された動作周波数が所定の下限値を下回ったことに応答して、プロセッサの動作周波数及び／又は電源電圧を該下限値に設定する下限値設定ステップを各プロセッサ毎にさらに備えることを特徴とする請求項 1 3 に記載のタスク管理方法。

【請求項 1 5】次の周期の開始までに実行を完了させる必要がある周期リアルタイム・タスクと実行完了時刻の制約がない非リアルタイム・タスクを並行して実行可能な複数のプロセッサによる演算処理の制御をコンピュータ・システム上で実行するように記述されたコンピュータ・ソフトウェアをコンピュータ可読形式で物理的に

格納した記憶媒体であって、前記コンピュータ・ソフトウェアは、

各プロセッサ毎に、稼動中の各時点において、各周期リアルタイム・タスクに課せられた要求を満たすために充分な動作周波数を算出する動作周波数算出ステップと、各プロセッサ毎に、前記動作周波数算出ステップによる算出結果に基づく動作周波数クロックを該プロセッサに供給するプロセッサ・クロック生成ステップと、各プロセッサ毎に、前記動作周波数算出ステップにより算出された動作周波数で該プロセッサを駆動させるために充分な電源電圧を算出する電源電圧算出ステップと、各プロセッサ毎に、前記電源電圧算出ステップによる算出結果に基づく電源電圧を該プロセッサに供給するプロセッサ電源供給ステップと、を具備することを特徴とする記憶媒体。

【請求項 1 6】次の周期の開始までに実行を完了させる必要がある周期リアルタイム・タスクと実行完了時刻の制約がない非リアルタイム・タスクを並行して実行可能な複数のプロセッサにおけるタスクの管理をコンピュータ・システム上で実行するように記述されたコンピュータ・ソフトウェアをコンピュータ可読形式で物理的に格納した記憶媒体であって、前記コンピュータ・ソフトウェアは、

各プロセッサ毎に、各周期リアルタイム・タスクに課せられた、次の周期の開始時間、周期、一周期あたりの処理負荷などの情報を、タスク実行前に記録するタスク情報記録ステップと、

各プロセッサ毎に、該プロセッサに課された処理負荷に応じて該プロセッサの電源電圧と動作周波数を設定する動作設定ステップと、

各プロセッサ毎に、前記タスク情報記録ステップにより記録された情報を参照して、電源電圧と動作周波数が該プロセッサの限界を越えず、及び／又は、既登録の周期リアルタイム・タスクに課せられた要求が満たされるように、新たな周期リアルタイム・タスクの登録を制限するタスク登録制御ステップと、を具備することを特徴とする記憶媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、1以上のタスクを同時に実行するタイプのプロセッサに関する電力削減技術に係り、特に、異なる周期で動作する1以上のタスクを実行するプロセッサに関する電力削減技術に関する。

【0002】更に詳しくは、本発明は、次の周期の開始までに実行を完了させる必要がある周期リアルタイム・タスクとかかる制約のない非リアルタイム・タスクを並行して実行するタイプのプロセッサに関する電力削減技術に係り、特に、マルチプロセッサ構成を採用することにより周期リアルタイム・タスク並びに非リアルタイム・タスク処理時におけるシステム全体の電力消費を削減

する電力削減技術に関する。

【0003】

【従来の技術】昨今のLSI (Large Scale Integration) 技術における革新的な進歩とも相俟って、各種の情報処理機器や情報通信機器が開発され、市販されるようになってきた。この種の機器では、CPU (Central Processing Unit) やその他のプロセッサが所定のプログラム・コードを実行することによりさまざまな処理サービスを提供するようになっている。

【0004】他方において、情報機器に関する消費電力の削減が最重要課題の1つとされている。これは、バッテリー駆動式の情報機器においてはバッテリー持続時間の延長に関わる問題だからである。また、商用電源で無尽蔵に駆動することができる情報機器においても、資源有限という社会生態学的な観点から省電力化が推奨されている。

【0005】情報機器内では、そのメイン・コントローラであるプロセッサの消費電力は、機器全体のそれに占める割合は高い。言い換えれば、プロセッサの省電力化は情報機器自体の省電力化にもつながる。一般には、プロセッサは、動作周波数の増大に従って演算速度が向上する一方で、消費電力が増大する傾向にある。

【0006】例えば、特開平11-194849号公報には、消費電力を無用に増加させることなく所定の処理時間に所定の処理動作を完了することができ、タスクの処理容量が変化する場合でも設定作業が簡単となるデータ処理方法及び装置について開示している。

【0007】同公報に開示されたデータ処理装置では、マイクロコンピュータが各種の処理動作を実行する場合の処理容量と処理時間を容量記憶手段と時間記憶手段とに登録しておき、マイクロコンピュータが各種の処理動作を実行する場合に対応する処理容量及び処理時間を選出し、処理容量を処理時間で除算してマイクロコンピュータの処理速度を算出して基準クロックの周波数を可変としている。マイクロコンピュータの処理速度を処理容量と処理時間に対応して可変するので、所定の処理動作を所定の処理時間に確実に完了することができる。また、基準クロックの周波数を最適値に設定できるので、データ処理装置における消費電力の無用な増加も防止できる。

【0008】しかしながら、同公報に開示されるデータ処理方法及び装置では、プロセッサの動作クロック周波数を変更するだけで消費電力の削減を図るものである。言い換えれば、動作クロック周波数の削減によって、単位時間あたりの消費電力は低下するものの、各処理を完了させるための所要時間が長くなり、この結果、総電力量を削減する効果はあまり高くない。すなわち、プロセッサがアイドル状態にあるときの消費電力量の範囲を越えず、効果として不充分である。

【0009】また、同公報に開示されるデータ処理方法

及び装置は、各処理の処理タイミングがあらかじめ定まっており、且つ、各処理を中断することなく順次処理することによってすべての処理を時間内に完了させることが可能なことを前提とするものである。このため、ある処理の実行を中断して、より緊急度が高い処理（例えばリアルタイム処理）を行わせる必要があるシステムに対しては適用することができない。

【0010】また、特開2000-122747号公報には、デジタル信号演算処理部にクロックを供給するクロック発生部を設けて、このクロック発生部からデジタル信号演算処理部へ供給するクロック周波数を、デジタル信号演算処理部での演算処理量に基づいて制御することによって消費電力を低減する制御装置及び方法について開示されている。

【0011】しかしながら、同公報に開示される制御装置及び方法では、演算部の動作クロック周波数を変更するだけで消費電力の削減を図るものである。言い換えれば、動作クロック周波数の削減によって、単位時間あたりの消費電力は低下するものの、各処理を完了させるための所要時間が長くなり、この結果、電力量の削減効果は、演算部がアイドル状態にあるときの消費電力量の範囲を越えず、効果として不充分である。

【0012】また、同公報に開示される制御装置及び方法では、アイドル時間が占める割合から動作周波数を算出するようになっている。ところが、異なる周期で動作するタスクが多数実行されるようなマルチタスク環境下では、アイドル時間が占める割合を計算することはできない。

【0013】また、Takanori Okuma, Tohru Ishihara, Hiroto Yasuura共著の論文“Real-Time Task Scheduling for a Variable Voltage Processor” (IEEE 12th International Symposium on System Synthesis, November 1999) において提案されるSS及びSDなるスケジューリング手法では、システムの稼動前に、タスクの実行開始時間が判っていることを前提としている。これは、タスクの追加や削除が行われる度に再度スケジューリングを行う必要があることを意味する。さらに、このスケジューリング処理は、各周期タスクの周期の最小公倍数を周期とするスケジューリングを計算することによって行わなければならない。それらの周期の最小公倍数が十分に小さくない場合、タスクの追加や削除の効率が悪化する。

【0014】また、この論文で提案されているDDスケジューリング手法は、周期スレッドのように、タスクが特定のパターンで起動されることを考慮したスケジューリングを行っていない。このため、消費電力の削減効果は不充分である。

【0015】また、Yann-Hang Lee, C. M. Krishna共著の論文“Voltage-Clock Scaling for Low Energy Consumption in Real-time Embedded Systems” (IEEE Sixth In

ternational Conference on Real-Time Computing Systems and Applications, December 1999) において提案されている "Task based static scheduling" なる手法は、タスクを静的優先度法でスケジューリングすることを前提としている。ところが、静的優先度スケジューリング法は最早デッドライン優先度スケジューリングに比べてスケジューリング能力が劣っていることで知られており、電力削減の効果として不十分である。

【0016】情報機器内のメイン・コントローラであるプロセッサの消費電力は、機器全体のそれに占める割合は高いので（前述）、従来は、消費電力の削減のために、システムに組み込むプロセッサ数を可能な限り減少させることが、当該技術分野における常識とされてきた。

【0017】一般には、プロセッサは、動作周波数の増大に従って演算速度が向上する一方で、消費電力が増大する傾向にある（前述）。また、プロセッサの動作周波数とともにその電源電圧（言い換えれば消費電力）をつり上げていかなければならない。（但し、実際には、LSI 製造プロセスの微細化によって電源電圧の上限が制限されているので、電圧を上げることによって周波数を上げることは行われない。）

【0018】例えば、プロセッサの動作周波数を 2 分の 1 に低下させた場合、その消費電力は 4 分の 1 まで低下することが知られている。このことは、同じ処理量のタスクを、単一のプロセッサによって実行するよりも、2 分の 1 の動作周波数で駆動する 2 個のプロセッサに処理を分散させた方が、処理時間が同じでしかも消費電力が 2 分の 1 になるということを意味する。

【0019】すなわち、複数のプロセッサに処理負荷を分散させた方がシステム全体の消費電力を削減することができる可能性があるが、これは消費電力の削減のためにプロセッサ数を可能な限り減少させるという従来から踏襲されてきた装置設計手法に合致しない結論に陥ってしまう。

【0020】プロセッサの動作周波数と電源電圧を動的制御により変化させることが可能なマルチプロセッサ構成のシステムであれば、各プロセッサに関して起動された各タスクを遅滞なく処理するために必要な動作周波数を適応的に変化させるとともに、時々刻々と切り替わる動作周波数に応じて最適な電源電圧を決定していくことで、各プロセッサ毎の消費電力を低減してシステム全体としても省電力化を達成することが可能と思料されるが、このようなことを実現した従来技術は見当たらない。

【0021】

【発明が解決しようとする課題】本発明の目的は、異なる周期で動作する 1 以上のタスクを実行する複数のプロセッサを備えたマルチプロセッサ構成システムのための、優れた電力削減技術を提供することにある。

【0022】本発明の更なる目的は、次の周期の開始までに実行を完了させる必要がある周期リアルタイムタスクとかかる制約のない非リアルタイムタスクを並行して実行するタイプのプロセッサを複数備えたマルチプロセッサ構成システムのための、優れた電力削減技術を提供することにある。

【0023】本発明の更なる目的は、アプリケーションのリアルタイム要求に応えつつプロセッサによる電力消費を削減することができる、マルチプロセッサ構成システムのための優れた電力削減技術を提供することにある。

【0024】本発明の更なる目的は、動作周波数と電源電圧を動的に制御することができるプロセッサを複数備えたマルチプロセッサ構成システムにおいて、各プロセッサ毎に、起動された各タスクを遅滞なく処理するために必要な動作周波数を適応的に変化させるとともに、時々刻々と切り替わる動作周波数に応じて最適な電源電圧を決定していくことで、各プロセッサ並びにシステム全体の消費電力を低減することができる、優れた電力削減技術を提供することにある。

【0025】

【課題を解決するための手段及び作用】本発明は、上記課題を参酌してなされたものであり、その第 1 の側面は、次の周期の開始までに実行を完了させる必要がある周期リアルタイム・タスクと実行完了時刻の制約がない非リアルタイム・タスクを並行して実行可能なプロセッサを複数個含んだ演算処理システム又は演算処理制御方法であって、各プロセッサ毎に、稼働中の各時点において、各周期リアルタイム・タスクに課せられた要求を満たすために充分な動作周波数を算出する動作周波数算出手段又はステップと、各プロセッサ毎に、前記動作周波数算出手段又はステップによる算出結果に基づく動作周波数クロックを該プロセッサに供給するプロセッサ・クロック生成手段又はステップと、各プロセッサ毎に、前記動作周波数算出手段又はステップにより算出された動作周波数で該プロセッサを駆動させるために充分な電源電圧を算出する電源電圧算出手段又はステップと、各プロセッサ毎に、前記電源電圧算出手段又はステップによる算出結果に基づく電源電圧を該プロセッサに供給するプロセッサ電源供給手段又はステップと、を具備することを特徴とする演算処理システム又は演算処理制御方法である。

【0026】但し、ここで言う「システム」とは、複数の装置（又は特定の機能を実現する機能モジュール）が論理的に集合した物のことを言い、各装置や機能モジュールが単一の筐体内にあるか否かは特に問わない。

【0027】プロセッサが実行するタスクの管理や、タスク実行に必要なプロセッサの動作周波数の算出は、例えば、オペレーティング・システムの機能として実装することができる。

【0028】本発明の第1の側面に係る演算処理システム及び演算処理制御方法によれば、各プロセッサ毎に、起動された周期リアルタイム・タスク並びに非リアルタイム・タスクを遅滞なく処理するために必要なプロセッサの動作周波数を適応的に変化させるとともに、時々刻々と切り替わる動作周波数に応じて最適なプロセッサ用電源電圧を決定していくことで、各プロセッサ毎の消費電力を低減することができる。

【0029】本発明の第1の側面に係る演算処理システム及び演算処理制御方法は、各プロセッサの稼働中の各時点において、与えられたタスクのうちいずれを実行すべきかを選択して実行するタスク選択・実行手段又はステップをさらに備えることができる。より具体的には、各プロセッサにおいて、実行可能な周期リアルタイム・タスクのリストの先頭からタスクを取り出し、該リストが空の場合には、実行可能な非リアルタイム・タスクのリストの先頭からタスクを取り出すようにすれば、各々のプロセッサは、周期リアルタイム・タスクに課された要求を満たしながら各タスクをその要求通りに実行することができる。

【0030】また、各プロセッサの稼働中の各時点において、実行すべきタスクがなくなったことに応答して、前記プロセッサを稼働率が低下したスリープ状態に移行させるスリープ遷移手段又はステップをさらに備えることで、未処理タスクがなくなったプロセッサをスリープ状態に遷移させることで、システム全体の消費電力を最大限に節約することができる。

【0031】また、本発明の第1の側面に係る演算処理システム及び演算処理制御方法は、各周期リアルタイム・タスクに課せられた、次の周期の開始時間、周期、一周期あたりの処理負荷などの情報を、タスク実行前に記録するタスク情報記録手段又はステップと、前記タスク情報記録手段又はステップにより記録された情報を参照して、電源電圧と動作周波数が前記プロセッサの限界を越えず、及び／又は、既登録の周期リアルタイム・タスクに課せられた要求を満たされるように、新たな周期リアルタイム・タスクの登録を制限するタスク登録制御手段又はステップと、を各プロセッサ毎にさらに備えるようにしてもよい。

【0032】また、各プロセッサ毎に、非リアルタイム・タスクが存在しないときに前記動作周波数算出手段又はステップによって算出された動作周波数が所定の下限値を下回ったことに応答して、あるいは、前記電源電圧算出手段又はステップによって算出された電源電圧が所定の下限値を下回ったことに応答して、該プロセッサの動作周波数を該下限値に設定する下限値設定手段又はステップをさらに備えるようにしてもよい。

【0033】このような場合、下限値に設定することによりプロセッサに生まれる余剰時間に非リアルタイム・タスクを実行することにより、周期リアルタイム・タ

クが次の周期の開始以前に実行を完了することを保証しつつ、非リアルタイム・タスクの実行によって消費される電力量を削減することができる。

【0034】また、本発明の第2の側面は、次の周期の開始までに実行を完了させる必要がある周期リアルタイム・タスクと実行完了時刻の制約がない非リアルタイム・タスクを並行して実行可能な複数のプロセッサのタスクを管理するタスク管理システム又は方法であって、各プロセッサ毎に、各周期リアルタイム・タスクに課せられた、次の周期の開始時間、周期、一周期あたりの処理負荷などの情報を、タスク実行前に記録するタスク情報記録手段又はステップと、各プロセッサ毎に、該プロセッサに課された処理負荷に応じて該プロセッサの電源電圧と動作周波数を設定する動作設定手段又はステップと、各プロセッサ毎に、前記タスク情報記録手段により記録された情報を参照して、電源電圧と動作周波数が該プロセッサの限界を越えず、及び／又は、既登録の周期リアルタイム・タスクに課せられた要求を満たされるように、新たな周期リアルタイム・タスクの登録を制限するタスク登録制御手段又はステップと、を具備することを特徴とするタスク管理システム又は方法である。

【0035】本発明の第2の側面に係るタスク管理システム又は方法によれば、電源電圧と動作周波数が前記プロセッサの限界を越えず、且つ、既登録の周期リアルタイム・タスクに課せられた要求を満たされるように、各プロセッサにおける新たな周期リアルタイム・タスクの登録を制限することができる。

【0036】また、非リアルタイム・タスクが存在しないときに前記動作周波数算出手段又はステップによって算出された動作周波数が所定の下限値を下回ったことに応答して、プロセッサの動作周波数及び／又は電源電圧を該下限値に設定する下限値設定手段又はステップを各プロセッサ毎にさらに備えるようにしてもよい。

【0037】このような場合、各プロセッサでは、下限値に設定することによりプロセッサに生まれる余剰時間に非リアルタイム・タスクを実行することにより、周期リアルタイム・タスクが次の周期の開始以前に実行を完了することを保証しながら、非リアルタイム・タスクの実行によって消費される電力量を削減することができる。

【0038】また、本発明の第3の側面は、次の周期の開始までに実行を完了させる必要がある周期リアルタイム・タスクと実行完了時刻の制約がない非リアルタイム・タスクを並行して実行可能な複数のプロセッサによる演算処理の制御をコンピュータ・システム上で実行するように記述されたコンピュータ・ソフトウェアをコンピュータ可読形式で物理的に格納した記憶媒体であって、前記コンピュータ・ソフトウェアは、各プロセッサ毎に、稼働中の各時点において、各周期リアルタイム・タスクに課せられた要求を満たすために十分な動作周波

数を算出する動作周波数算出ステップと、各プロセッサ毎に、前記動作周波数算出ステップによる算出結果に基づく動作周波数クロックを該プロセッサに供給するプロセッサ・クロック生成ステップと、各プロセッサ毎に、前記動作周波数算出ステップにより算出された動作周波数で該プロセッサを駆動させるために充分な電源電圧を算出する電源電圧算出ステップと、各プロセッサ毎に、前記電源電圧算出ステップによる算出結果に基づく電源電圧を該プロセッサに供給するプロセッサ電源供給ステップと、を具備することを特徴とする記憶媒体である。

【0039】また、本発明の第4の側面は、次の周期の開始までに実行を完了させる必要がある周期リアルタイム・タスクと実行完了時刻の制約がない非リアルタイム・タスクを並行して実行可能な複数のプロセッサにおけるタスクの管理をコンピュータ・システム上で実行するように記述されたコンピュータ・ソフトウェアをコンピュータ可読形式で物理的に格納した記憶媒体であって、前記コンピュータ・ソフトウェアは、各プロセッサ毎に、各周期リアルタイム・タスクに課せられた、次の周期の開始時間、周期、一周あたり処理の処理負荷などの情報を、タスク実行前に記録するタスク情報記録ステップと、各プロセッサ毎に、該プロセッサに課せられた処理負荷に応じて該プロセッサの電源電圧と動作周波数を設定する動作設定ステップと、各プロセッサ毎に、前記タスク情報記録ステップにより記録された情報を参照して、電源電圧と動作周波数が該プロセッサの限界を越えず、及び／又は、既登録の周期リアルタイム・タスクに課せられた要求が満たされるように、新たな周期リアルタイム・タスクの登録を制限するタスク登録制御ステップと、を具備することを特徴とする記憶媒体である。

【0040】本発明の第3及び第4の各側面に係る記憶媒体は、例えば、様々なプログラム・コードを実行可能な汎用性のコンピュータ・システムに対して、コンピュータ・ソフトウェアをコンピュータ可読な形式で物理的に提供する媒体である。このような媒体は、例えば、CD (Compact Disc) やFD (Floppy Disc)、MO (Magnet-Optical disc) などの着脱自在で可搬性の記憶媒体である。あるいは、ネットワーク（ネットワークは無線、有線の区別を問わない）などの伝送媒体などを經由してコンピュータ・ソフトウェアを特定のコンピュータ・システムにコンピュータ可読形式で提供することも技術的に可能である。

【0041】このような記憶媒体は、コンピュータ・システム上で所定のコンピュータ・ソフトウェアの機能を実現するための、コンピュータ・ソフトウェアと記憶媒体との構造上又は機能上の協働的關係を定義したものである。換言すれば、本発明の第3及び第6の各側面に係る記憶媒体を介して所定のコンピュータ・ソフトウェアをコンピュータ・システムにインストールすることによって、コンピュータ・システム上では協働的作用が発揮

され、本発明の第1及び第2の各側面に係る演算処理システム及び演算処理制御方法、タスク管理システム及びタスク管理方法と同様の作用効果を得ることができる。

【0042】本発明のさらに他の目的、特徴や利点は、後述する本発明の実施例や添付する図面に基づくより詳細な説明によって明らかになるであろう。

【0043】

【発明の実施の形態】以下、図面を参照しながら本発明の実施例を詳解する。

10 【0044】1. システム構成

図1には、本発明の実施に供されるマルチプロセッサ構成の演算処理システム10のハードウェア構成を模式的に示している。同図に示すように、演算処理システム10は、プロセッサ11-A及び11-Bと、RAM (Random Access Memory) 12と、ROM (Read Only Memory) 13と、周辺デバイス14と、タイマ15とを含んでいる。同図に示す演算処理システム10は、プロセッサ11は2個しか装備していないが、3個以上のプロセッサを備えるマルチプロセッサ構成であっても、同様に本発明の効果を実証することを理解されたい。

【0045】プロセッサ11-A及び11-Bは、演算処理システム10のメイン・コントローラであり、それぞれオペレーティング・システム (OS) の制御下で、各種のプログラム・コードを実行するようになっている。

【0046】オペレーティング・システムがプログラム実行を管理・制御する単位は、一般に「タスク」と呼ばれる。本実施例に係るプロセッサ11-A及び11-Bは、異なる周期で動作する複数のタスクを同時に実行するマルチタスク機構を備えているものとする。タスクには、次の周期の開始までに実行を完了させる必要がある「周期リアルタイム・タスク」と、このような実行完了時間に制約がない「非リアルタイム・タスク」とに大別することができる。

【0047】プロセッサ11-A及び11-Bは、バス16によって他の機器類（後述）と相互接続されている。バス16上の各機器にはそれぞれ固有のメモリ・アドレス又はI/Oアドレスが付与されており、プロセッサ11-A及び11-Bはこれらアドレスによって特定の機器へのアクセスが可能となっている。バス16は、アドレス・バス、データ・バス、コントロール・バスなどを含む共通信号伝送路である。

【0048】RAM12は、書き込み可能なメモリであり、プロセッサ11において実行されるプログラム・コードをロードしたり、実行プログラムの作業データを一時格納するために使用される。プログラム・コードには、例えば、BIOS (Basic Input/Output System: 基本入出力システム)、周辺機器をハードウェア操作するためのデバイス・ドライバ、オペレーティング・システム、アプリケーションなどが挙げられる。

【0049】ROM13は、所定のコードやデータを恒久的に記憶するための不揮発メモリであり、例えば、BIOSや始動時の自己診断プログラム（Power On Self Test：POST）などを格納している。

【0050】周辺デバイス14には、ディスプレイやプリンタのようなユーザ出力装置、キーボードやマウスのようなユーザ入力装置、ハード・ディスクやその他のメディア・ドライブからなる外部記憶装置、ネットワーク・インターフェース・カードのような通信装置が含まれる。

【0051】タイマ15は、タイマ信号を所定周期で発生させる装置である。タイマ15にも割り込みレベルが割り当てられており、プロセッサ11に対して周期的な割り込みを発生するようになっている。（但し、周期の異なる複数の周期リアルタイム・タスクが存在する場合、タイマ信号は周期的な割り込みにはならない。）

【0052】上述したようなシステム10の各コンポーネントには、電源装置17からの電力が電源供給線18を介して供給される。電源装置17は、例えばバッテリーや商用AC電源で構成されるが、AC/DCアダプタやDC/DCコンバータによって一定の電源電圧を供給することができる。

【0053】図示の例では、プロセッサ11-Aに対しては、専用のDC/DC変換器21-Aが配設されている。プロセッサ11-Aは、オペレーティング・システムの制御下で、プロセッサ用DC/DC変換器21-Aからの供給電圧を設定する機構を備えている。

【0054】また、プロセッサ11-Aは、プロセッサ・クロック生成器22が発生する動作クロックを入力して、その動作周波数に同期的に駆動する。一般には、動作周波数の増大により、プロセッサ11-Aの処理速度は向上するとともに消費電力も増大する。例えば、プロセッサ11-Aの動作周波数を2分の1に低下させるとその消費電力は4分の1になる。本実施例では、プロセッサ11-Aは、オペレーティング・システムの制御下で、プロセッサ・クロック生成器22-Aが生成するクロックの動作周波数を設定する機構を備えている。

【0055】また、図1に示すように、他方のプロセッサ11-Bに対しても、同様に、プロセッサ用DC/DC変換器21-B並びにプロセッサ・クロック生成器22-Bが装備されており、プロセッサ11-Bはオペレーティング・システムの制御下で自身への電源電圧やクロックの動作周波数を設定することができるようになっている。

【0056】なお、プロセッサ11に対する電源電圧と動作周波数の双方をプロセッサ用DC/DC変換器21とプロセッサ・クロック生成器22の各々によって動的に制御する必要は必ずしもなく、いずれか一方の動作によっても本発明の効果を実現することができる。言い換えれば、オペレーティング・システムは、プロセッサ1

1の電源電圧と動作周波数の双方を動的制御するのではなく、いずれか一方のみを制御する場合であっても本発明の効果を実現することができる。また、電源電圧と動作周波数のいずれか一方のみを演算処理により設定し、他方はその設定に自動的に追従するように構成してもよい（例えば、プロセッサ11の周波数をオペレーティング・システムが設定することによって、その周波数で動作するために必要な最小の電源電圧が自動的にプロセッサ11に供給されるように構成してもよい）。また、複数のプロセッサのうち一部に対してのみ電源電圧と動作周波数の動的制御を行うようにしても、同様に本発明の効果を奏することを理解されたい。

【0057】2. アプリケーションに対するインターフェース

演算処理システム10上における演算処理は、各プロセッサ11がオペレーティング・システムの制御下でアプリケーション・タスクを実行するという形式で実現される（前述）。オペレーティング・システムとは、システム10のハードウェア及びソフトウェアを総合的に管理するための基本ソフトウェアのことであり、アプリケーションに対しては、オペレーティング・システムの基本機能呼び出す（コールする）ためのインターフェース、すなわちAPI（Application Programming Interface）を提供している。

【0058】オペレーティング・システムがアプリケーションの実行を管理・制御する単位は「タスク」と呼ばれ、タスクには、次の周期の開始までに実行を完了させる必要がある「周期リアルタイム・タスク」と、このような実行完了時間に制約がない「非リアルタイム・タスク」とに大別される（前述）。

【0059】周期リアルタイム・タスクとは、ある周期で定期的に起動されるタスクである。但し、各リアルタイム・タスクの周期は区々である。図2には、複数の周期リアルタイム・タスクが異なる周期で同時に実行されている様子を模式的に図解している。

【0060】周期リアルタイム・タスクは、各起動毎に、あらかじめアプリケーション・プログラムによって設定されたプロセッサ使用量を越えない範囲で実行が行われる。ここで言う「プロセッサ使用量」とは、プロセッサの動作周波数が最大値に固定された条件下で、一周の実行を完了させるために必要な時間のことを意味する。

【0061】本実施例に係るオペレーティング・システムは、周期リアルタイム・タスクの実行が次の周期が開始する以前に完了するように動作周波数の制御とタスク実行順序の制御を行うようになっている。

【0062】また、非リアルタイム・タスクとは、その実行完了時間に対する制約がないタスクである。

【0063】本実施例に係るオペレーティング・システムは、アプリケーション・プログラムに対して、「周期

リアルタイム・タスクの登録」、「周期リアルタイム・タスクの周期的実行」、「周期リアルタイム・タスクの登録解除」、「非リアルタイム・タスクの実行開始」、「非リアルタイム・タスクの実行完了」、「動作周波数の下限値の設定」というインターフェースを提供している。

【0064】(1) 周期リアルタイム・タスクの登録アプリケーション・プログラムは、このインターフェースを利用する際に、以下の各情報をオペレーティング・システムに与える。

タスクの起動周期：タスクが起動される周期である。
タスクの一周期あたりの処理負荷：処理負荷の表現方法として様々なものを挙げることができる。例えば、プロセッサが最大動作周波数でこのタスクを実行し続けるときに、一周期の処理に要する計算時間で表現することも可能である。

タスクの実行に必要なその他の情報：例えば、タスクの実行開始アドレスやスタック領域の位置などを与えることが挙げられる。但し、この情報は、動作周波数や電源電圧の制御やタスク実行順序の制御には本質的なものではない。

【0065】オペレーティング・システムは、これらの情報を基に、既に登録された周期リアルタイム・タスクの要求を満たしつつ、新たに周期リアルタイム・タスクの実行がさらに可能か否かを判断する。実行不可能と判断したときには、アプリケーション・プログラムに対してその旨を通知するとともに、タスクの登録を行わない。他方、実行可能と判断したときには、タスクの登録を行う。

【0066】(2) 周期リアルタイム・タスクの周期的実行

上述のインターフェース「周期リアルタイム・タスクの登録」によって登録に成功した周期リアルタイム・タスクは、このインターフェースの呼び出しによって周期的な起動を開始することができる。アプリケーション・プログラムは、このインターフェースを利用する際に、タスクを特定できる値をオペレーティング・システムに与える。

【0067】なお、このインターフェースを省略して、タスクの登録成功と同時に起動を開始するように実装することも可能である。また、指定された時間から周期的な起動を始めることを要求するためのインターフェースを用意することによっても実装可能である。

【0068】(3) 周期リアルタイム・タスクの登録解除

アプリケーション・プログラムは、このインターフェースを利用する際に、タスクを特定できる値をオペレーティング・システムに与える。

【0069】(4) 非リアルタイム・タスクの実行開始
このインターフェースの呼び出しによって、指定された

タスクは実行可能状態となる。アプリケーション・プログラムは、このインターフェースを利用する際に、タスクの実行開始アドレスやスタック領域の位置、さらに非リアルタイム・タスク間の実行優先順位などの情報を、オペレーティング・システムに与える。

【0070】(5) 非リアルタイム・タスクの実行完了
このインターフェースを呼び出すことによって、指定されたタスクは以後実行されなくなる。但し、上述の「非リアルタイム・タスクの実行開始」インターフェースの呼び出しによって、再度実行を開始することができるように実装することも可能である。

【0071】(6) 動作周波数の下限値の設定
オペレーティング・システムによって設定される動作周波数が、ある閾値以下に低下しないように設定するためのインターフェースである。オペレーティング・システムは、通常、周期リアルタイム・タスクを適切に実行するのに充分な必要最低限の動作周波数でプロセッサを動作させようとする。このため、非リアルタイム・タスクの実行に費やされるプロセッサ時間が不充分となることがある。このような場合には、この閾値を大きく設定することによって、非リアルタイム・タスクを実行するだけの余地を与えることができる。

【0072】3. オペレーティング・システムの実装
本実施例に係るオペレーティング・システムは、各タスク毎に、以下の変数を保持している。但し、各変数の添え字 i はタスクを識別するために与えられるタスク番号であるとす。

【0073】 e_i ：次の周期の開始時間

p_i ：周期

c_i ：一周期あたりの処理負荷

【0074】さらに、本実施例に係るオペレーティング・システムは以下の変数を保持している。

【0075】 r_p ：現在、 p 番目のプロセッサ p に登録されている周期リアルタイム・タスクの処理負荷を周期で割った値の合計値である。すなわち、プロセッサ p の使用率に相当し、0 から 1.0 の値をとる。 r_p が 1.0 未満の値であれば、プロセッサ p においてさらに周期リアルタイム・タスクを実行可能な状態である。 r_p は、プロセッサ p において周期リアルタイム・タスクを登録するときに更新される値である。

【0076】 l_p ：現在、 p 番目のプロセッサ p に登録されており、且つ、起動要求が発行された周期リアルタイム・タスクの処理負荷を周期で割った値の合計値である。すなわち、プロセッサ p の使用率に相当し、0 から 1.0 の値をとる。 l_p は、プロセッサ p において周期リアルタイム・タスクを起動するときに逐次更新される値である。

【0077】 $f_{min,p}$ ： p 番目のプロセッサ p についての、アプリケーション・プログラムによって設定された動作周波数の下限値である。

【0078】 $R_{r,p}$: p 番目のプロセッサ p についての、実行可能な周期リアルタイム・タスクのリストである。このリストは、ある周期において起動された後に、いまだその周期の実行を完了しておらず、且つ、様々なイベントの発生を待っていない状態にあるすべての周期リアルタイム・タスクで構成される。このリスト中では、各タスクは、その e_i (次の周期の開始時間) の値が小さいものから順に並べられている。

【0079】 $R_{b,p}$: p 番目のプロセッサ p についての、実行可能な非リアルタイム・タスクのリストである。このリストは、タスクが起動された後に、いまだ実行を完了しておらず、且つ、様々なイベントの発生を待っていない状態にあるすべての非リアルタイム・タスクで構成される。このリスト中のタスクの順序は任意でよいが (例えば先入れ先出し (FIFO) 方式でもよい)、非リアルタイム・タスクに実行優先順位を設定するような場合には、その優先順位に応じた順序で並べるようにしてもよい。

【0080】図3には、本実施例に係るオペレーティング・システムの機能構成を、特にタスク管理機能に着目して図解している。

【0081】図示の通り、各プロセッサ 11-A, 11-B 上で動作するそれぞれのオペレーティング・システムは、スケジューラとタイマ・マネージャを含んでいる。

【0082】タイマ・マネージャは、タイマ 15 から供給されるタイマ信号に従ってプロセッサ 11 をタイマ管理する機能モジュールである。タイマ・マネージャは、要求された各タスクを所定の順序 (例えば、実行開始時刻に従った順序) で保持する待ち行列 (queue) である。待ち行列中の各タスクは、タスク識別子と実行開始時刻 (release time) を含んでいる。

【0083】図示の例では、プロセッサ 11-A 側におけるタイマ・マネージャ内の待ち行列には、タスク 10 をリリース時刻 80 で要求する要求 1 と、タスク 11 をリリース時刻 95 で要求する要求 2 が格納されている。

【0084】スケジューラは、起動された各タスクのスケジュール管理を行う機能モジュールであり、実行可能なリアルタイム・タスクのリスト R_r と、実行可能な非リアルタイム・タスクのリスト R_b を含んでいる。また、スケジューラは、周期リアルタイム・タスクの処理負荷を周期で割った値の合計値 r 並びに l を管理している (r は周期リアルタイム・タスク登録時の値であり、 l は周期リアルタイム・タスク起動要求時の値である)。

【0085】図示の例では、リスト R_r には、タスク 1、タスク 3、タスク 4、タスク 2 の順序で周期リアルタイム・タスクが登録されている。また、リスト R_b には、タスク 6、タスク 8、タスク 7、タスク 5 の順序で非リアルタイム・タスクが登録されている。また、処理

負荷を周期で割った値の合計値 r 及び l はともに 80% を示し、プロセッサがさらに周期リアルタイム・タスクを実行可能な状態であることを示している。

【0086】タイマ・マネージャは、スケジューラに対して、タスクの起動すなわちリリースを通知する。また、スケジューラは、タイマ・マネージャに対して、タスク番号とタスク起動時刻を通知する。

【0087】アプリケーション・プログラムは、オペレーティング・システム内のスケジューラに対して、周期リアルタイムタスクの登録 (p_i , c_i の通知を伴う)、周期リアルタイム・タスクの起動・登録削除、非リアルタイム・タスクの登録並びに登録削除などの要求を、上述した各インターフェースによって受け付けるようになっている。

【0088】また、プロセッサ 11-A 及び 11-B で動作するオペレーティング・システムの間では、各変数を参照し合う通信手段を備えている。

【0089】4. オペレーティング・システムによるタスク制御

続いて、上述したオペレーティング・システムの各インターフェースを実装したプログラムによる処理動作について説明する。

【0090】図4には、周期リアルタイム・タスクを登録するインターフェースにおいて実現される処理手順をフローチャートの形式で示している。以下、このフローチャートに従って周期リアルタイム・タスクの登録処理について説明する。

【0091】まず、各プロセッサに問い合わせて、周期リアルタイム・タスクの処理負荷を周期で割った値の合計値 r_p が最小となるプロセッサ p を探索する (ステップ S1)。

【0092】そして、プロセッサ p において、タスク登録判定処理を実行する (ステップ S2)。

【0093】ステップ S2 におけるタスク登録判定処理は、別途定義済みの処理ルーチンであり、その詳細な処理手順は図5においてフローチャートの形式で示している。以下、図5を参照しながら周期リアルタイム・タスクの登録判定処理について説明する。

【0094】まず、登録判定の対象となる周期リアルタイム・タスクにタスク番号 i を付与するとともに、登録判定の対象となるプロセッサを p とする (ステップ S11)。

【0095】そして、登録判定の対象となるタスク i を含めて、現在オペレーティング・システムに登録されている周期リアルタイム・タスクの処理負荷を周期で割った値の合計値 r_p ($= c_i / p_i + r_p$) を試算してみて、これが 1 未満か否かをチェックする (ステップ S12)。

【0096】プロセッサ p の処理負荷に相当する合計値 r_p が 1 を越える場合には、もはやシステム 10 は周期

リアルタイム・タスク i を実行することができない状態なので、登録失敗とする。

【0097】他方、試算した合計値 r_p が 1 未満である場合には、システム 10 はさらにタスク i を追加して実行することができる状態なので、 r_p の値を更新して

(ステップ S13)、登録成功とする。

【0098】図 6 には、周期リアルタイム・タスクを登録解除するインターフェースにおいて実現される処理手順をフローチャートの形式で示している。以下、このフローチャートに従って、周期リアルタイム・タスクの登録解除処理について説明する。

【0099】まず、周期リアルタイム・タスクの登録解除要求を受け取ったプロセッサ p は、該タスクが自分に割り当てられたものか否かを、スケジューラ又は待ち行列で確認する (ステップ S21)。

【0100】タスクがプロセッサ p 自身のものである場合には、登録削除すべき周期リアルタイム・タスクにタスク番号 i を付与する (ステップ S22)。

【0101】次いで、現在オペレーティング・システムに登録されている周期リアルタイム・タスクの処理負荷を周期で割った値の合計値 r_p 、並びに、現在オペレーティング・システムに登録され且つ起動要求が発行された周期リアルタイム・タスクの処理負荷を周期で割った値の合計値 l_p の値から、登録削除を行う周期リアルタイムタスク i の分 c_i/p_i を取り除く (ステップ S23, S24)。

【0102】また、登録削除を行う周期リアルタイム・タスク i が既にスケジューラ内の実行可能リアルタイム・タスクのリスト $R_{r,p}$ 中に登録されている場合には、タスク i をリスト $R_{r,p}$ から削除する (ステップ S25)。

【0103】そして、別途定義された再スケジューリング処理を行うことで (ステップ S26)、本処理ルーチン全体を終了する。再スケジューリング処理については後に詳解する。

【0104】他方、プロセッサ p 自身のものではない場合には、他のプロセッサへ登録解除要求を転送して (ステップ S27)、該要求の処理完了を待って (ステップ S28)、本処理ルーチン全体を終了する。

【0105】図 7 には、周期リアルタイム・タスクを起動開始するインターフェースにおいて実現される処理手順をフローチャートの形式で示している。以下、このフローチャートに従って、 p 番目のプロセッサ p において周期リアルタイム・タスクの起動開始処理について説明する。

【0106】まず、起動開始すべき周期リアルタイム・タスクにタスク番号 i を付与し (ステップ S31)、現在オペレーティング・システムに登録され且つ起動要求が発行された周期リアルタイム・タスクの処理負荷を周期で割った値に合計値 l_p にタスク i の分 c_i/p_i を加

算する (ステップ S32)。

【0107】次いで、現在オペレーティング・システムに登録され且つ起動要求が発行された周期リアルタイム・タスクの処理負荷を周期で割った値に合計値 l_p と、アプリケーション・プログラムによって設定された動作周波数の下限値 $f_{min,p}$ のうち、大きい値を変数 f に設定する (ステップ S33)。

【0108】そして、プロセッサ p の動作周波数を f に設定するとともに、プロセッサ p に供給する電源電圧を周波数 f で動作可能な最小の電圧に設定する (ステップ S34)。プロセッサ p の動作周波数や電源電圧の変更は、プロセッサ・クロック生成器 22 並びにプロセッサ用 DC/DC 変換器 21 の各々に指示値を送信することによって行われる (前述)。

【0109】プロセッサ p に課された処理負荷から求まる動作周波数がアプリケーション・プログラムによって設定された動作周波数の下限値 f_{min} を下回る場合には、下限値 $f_{min,p}$ を動作周波数として設定することにより、プロセッサ p に生まれる余剰時間を利用して非リアルタイム・タスクを実行することができる。この結果、周期リアルタイム・タスクが次の周期の開始以前に実行を完了することを保証しつつ、非リアルタイム・タスクの実行によって消費される電力量を削減することができる。

【0110】次いで、現在時刻にタスク i が持つ周期 p_i を加算した値を次の周期の開始時刻 e_i に代入して (ステップ S35)、時刻 e_i にタスク i が起動するようにタイマ 15 を設定する (ステップ S36)。

【0111】次いで、スケジューラ内の実行可能リアルタイム・タスクのリスト $R_{r,p}$ 中にタスク i を追加登録する (ステップ S37)。

【0112】そして、別途定義された再スケジューリング処理を行うことで (ステップ S38)、本処理ルーチン全体を終了する。再スケジューリング処理については後に詳解する。

【0113】図 8 には、プロセッサ p において周期リアルタイム・タスクを起動するタイマの設定時刻に達したときにおける処理手順をフローチャートの形式で示している。以下、このフローチャートに従って、周期リアルタイム・タスクに指定された時刻に達したときの処理について説明する。

【0114】まず、起動すべき周期リアルタイム・タスクにタスク番号 i を付与する (ステップ S41)。

【0115】次いで、現在時刻にタスク i が持つ周期 p_i を加算した値を次の周期の開始時刻 e_i に代入して (ステップ S42)、時刻 e_i にタスク i が起動するようにタイマ 15 を設定する (ステップ S43)。

【0116】次いで、プロセッサ p におけるスケジューラ内の実行可能リアルタイム・タスクのリスト $R_{r,p}$ 中にタスク i を追加登録する (ステップ S44)。

【0117】そして、別途定義された再スケジューリング処理を行うことで（ステップS45）、本処理ルーチン全体を終了する。再スケジューリング処理については後に詳解する。

【0118】図9には、プロセッサpにおいて非リアルタイム・タスクを登録するインターフェースにおいて実現される処理手順をフローチャートの形式で示している。以下、このフローチャートに従って非リアルタイム・タスクの登録処理について説明する。

【0119】まず、新規登録する非リアルタイム・タスクにタスク番号iを付与する（ステップS51）。そして、このタスクiを、プロセッサpにおけるスケジューラ内の実行可能な非リアルタイム・タスクのリストR_{b,p}に追加登録する（ステップS52）。

【0120】そして、別途定義された再スケジューリング処理を行うことで（ステップS53）、本処理ルーチン全体を終了する。再スケジューリング処理については後に詳解する。

【0121】図10には、プロセッサpにおいて非リアルタイム・タスクを登録削除するインターフェースにおいて実現される処理手順をフローチャートの形式で示している。以下、このフローチャートに従って、非リアルタイム・タスクの登録削除処理について説明する。

【0122】まず、登録削除する非リアルタイム・タスクにタスク番号iを付与する（ステップS61）。

【0123】このタスクiが、プロセッサpのスケジューラ内の実行可能な非リアルタイム・タスクのリストR_{b,p}に既に登録されている場合には、該リストR_{b,p}中から削除する（ステップS62）。

【0124】そして、別途定義された再スケジューリング処理を行うことで（ステップS63）、本処理ルーチン全体を終了する。再スケジューリング処理については後に詳解する。

【0125】図11には、プロセッサpにおいて周期リアルタイム・タスクの登録削除時、起動開始時、非リアルタイムタスクの登録時、登録削除時の各々の処理ルーチン内で行われる再スケジューリング処理の手順をフローチャートの形式で示している。以下、このフローチャートを参照しながら、再スケジューリング処理について説明する。

【0126】まず、プロセッサpにおけるスケジューラ内の実行可能な周期リアルタイム・タスクのリストR_{r,p}が空か否かをチェックする（ステップS71）。

【0127】リストR_{r,p}が空でなければ、該リストR_{r,p}中の先頭の周期リアルタイム・タスクにタスク番号iを付与して（ステップS74）、タスクiに制御を移し（ステップS76）、本処理ルーチン全体を終了する。

【0128】リストR_{r,p}が空の場合、さらに、プロセッサpにおけるスケジューラ内の実行可能な非リアルタイム・タスクのリストR_{b,p}が空か否かをチェックする（ステップS72）。

【0129】リストR_{b,p}が空でなければ、該リストR_{b,p}中の先頭の非リアルタイム・タスクにタスク番号iを付与して（ステップS75）、タスクiに制御を移し（ステップS76）、本処理ルーチン全体を終了する。

【0130】スケジューラ内のリストR_{r,p}及びリストR_{b,p}のいずれも空である場合には、現在プロセッサpが実行すべきタスクがないことになるので、プロセッサpをスリープ状態に移行させ（ステップS73）、次のタスクが発生するまでプロセッサpを待機せしめる。

【0131】なお、ここで言うスリープ状態とは、プロセッサ11の活動を低下させて省電力化を図る動作モードのことを指す。但し、スリープ状態の定義については本発明の要旨とは直接関連しないので、ここでは敢えて説明しない。

【0132】〔追補〕以上、特定の実施例を参照しながら、本発明について詳解してきた。しかしながら、本発明の要旨を逸脱しない範囲で当業者が該実施例の修正や代用を成し得ることは自明である。

【0133】本明細書で紹介した実施の形態では、周期リアルタイム・タスクはタイマで起動されるものとしたが、本発明の要旨は必ずしもこれに限定されない。最小起動間隔が指定の周期を下回らない範囲で、自由な時刻に周期リアルタイム・タスクを起動するようにしてもよい。例えば、ディスプレイ装置を備えた演算処理システムにおいて、垂直帰線期間割り込み信号が発生したときに、タスクを起動することが考えられる。また、ネットワークからのパケット到着時に周期リアルタイム・タスクを起動することも可能である。

【0134】要するに、例示という形態で本発明を開示してきたのであり、限定的に解釈されるべきではない。本発明の要旨を判断するためには、冒頭に記載した特許請求の範囲の欄を参酌すべきである。

【0135】

【発明の効果】以上詳記したように、本発明によれば、異なる周期で動作する1以上のタスクを実行する複数のプロセッサを備えたマルチプロセッサ構成システムのための、優れた電力削減技術を提供することができる。

【0136】また、本発明によれば、次の周期の開始までに実行を完了させる必要がある周期リアルタイムタスクとかかる制約のない非リアルタイムタスクを並行して実行するタイプのプロセッサを複数備えたマルチプロセッサ構成システムのための、優れた電力削減技術を提供することができる。

【0137】また、本発明によれば、アプリケーションのリアルタイム要求に応えつつプロセッサによる電力消費を削減することができる、マルチプロセッサ構成システムのための優れた電力削減技術を提供することができる。

【0138】また、本発明によれば、動作周波数と電源電圧を動的に制御することができるプロセッサを複数備えたマルチプロセッサ構成システムにおいて、各プロセッサ毎に、起動された各タスクを遅滞なく処理するために必要な動作周波数を適応的に変化させるとともに、時々刻々と切り替わる動作周波数に応じて最適な電源電圧を決定していくことで、各プロセッサ並びにシステム全体の消費電力を低減することができる、優れた電力削減技術を提供することができる。

【0139】また、本発明に係るマルチプロセッサ構成システムを利用した場合、単一のプロセッサにより同じ処理量のタスクを実行する場合に比し、消費電力を削減することができる。

【0140】また、本発明によれば、プロセッサの動作周波数や電源電圧の変更頻度を低く抑えながら、各々のプロセッサによる電力消費を削減することができる。

【0141】周期リアルタイム・タスクは次の周期の開始以前に処理を完了するタスクである。処理を完了すべき時間すなわちデッドラインを自由に設定可能なようにシステムを拡張することも考えられるが、本発明では敢えてこのように処理完了時間に制限を課すことによって、タスクの登録や登録削除時に動作周波数を変化させるだけで充分に高い電力削減効果を得ることができる。

【0142】したがって、プロセッサの電源電圧や動作周波数を変化させるために比較的長い時間を要するハードウェアに対しても、本発明を適用することができる。

【0143】例えば、あるタスク i について、デッドラインを次の周期の開始時間よりも前に設定することがシステム構築上どうしても必要になった場合に、タスク起動タイマの時刻設定を除く、変数 p_i に対するすべての参照をデッドラインに対する参照へと置き換えるだけでよい。

【図面の簡単な説明】

【図1】本発明の実施に供されるマルチプロセッサ構成の演算処理システム10のハードウェア構成を模式的に示した図である。

【図2】複数の周期リアルタイム・タスクが異なる周期で同時に実行されている様子を模式的に示した図である。

【図3】本実施例に係るオペレーティング・システムの機能構成を模式的に示したブロック図である。

【図4】周期リアルタイム・タスクを登録するインターフェースにおいて実現される処理手順を示したフローチャートである。

【図5】プロセッサ p における周期リアルタイム・タスクの登録判定処理の手順を示したフローチャートである。

【図6】周期リアルタイム・タスクを登録解除するインターフェースにおいて実現される処理手順を示したフローチャートである。

【図7】周期リアルタイム・タスクを起動開始するインターフェースにおいて実現される処理手順を示したフローチャートである。

【図8】周期リアルタイム・タスクを起動するタイマの設定時刻に達したときにおける処理手順を示したフローチャートである。

【図9】非リアルタイム・タスクを登録するインターフェースにおいて実現される処理手順を示したフローチャートである。

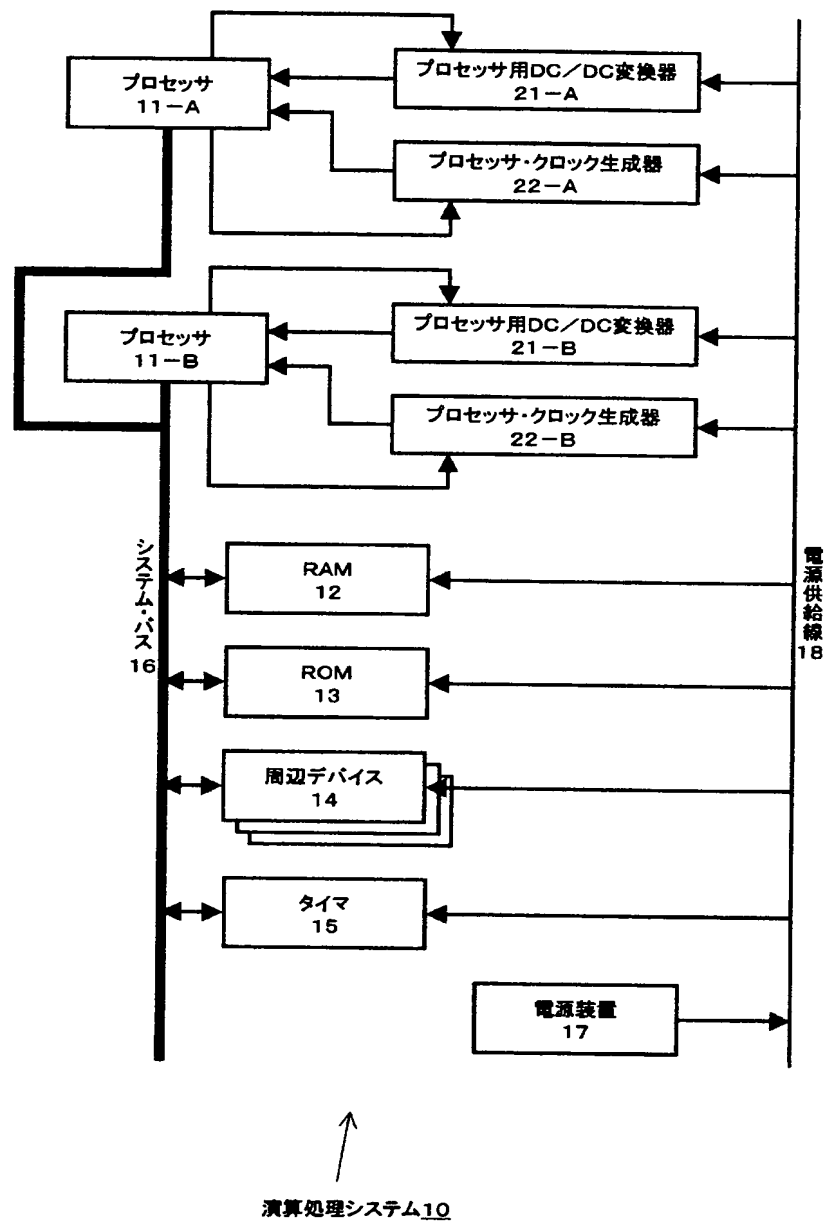
【図10】非リアルタイム・タスクを登録削除するインターフェースにおいて実現される処理手順を示したフローチャートである。

【図11】プロセッサ p において周期リアルタイム・タスクの登録削除時、起動開始時、非リアルタイムタスクの登録時、登録削除時に行う再スケジューリング処理の手順を示したフローチャートである。

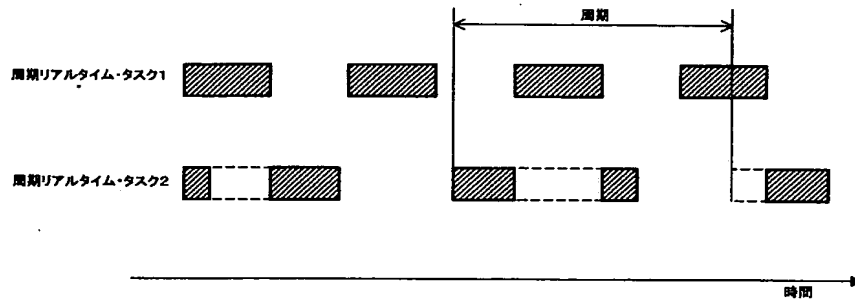
【符号の説明】

- 10…演算処理システム
- 11…プロセッサ
- 12…RAM
- 13…ROM
- 14…周辺デバイス
- 15…タイマ
- 16…システム・バス
- 17…電源装置
- 18…電源供給線
- 21…プロセッサ用DC/DC変換器
- 22…プロセッサ・クロック生成器

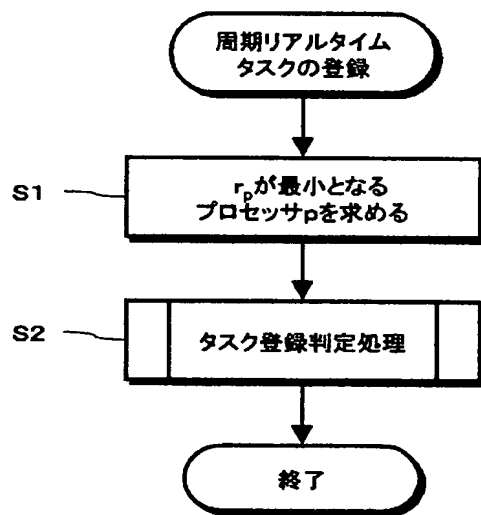
【図1】



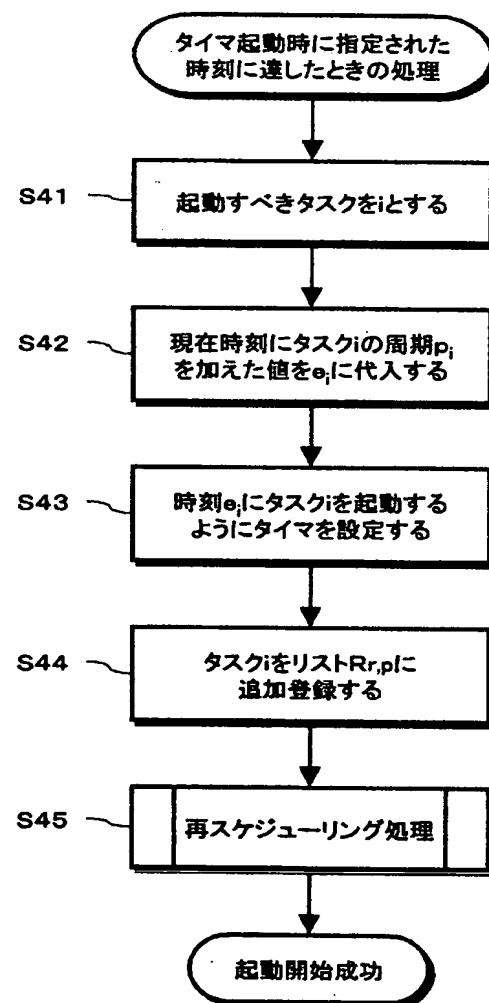
【図2】



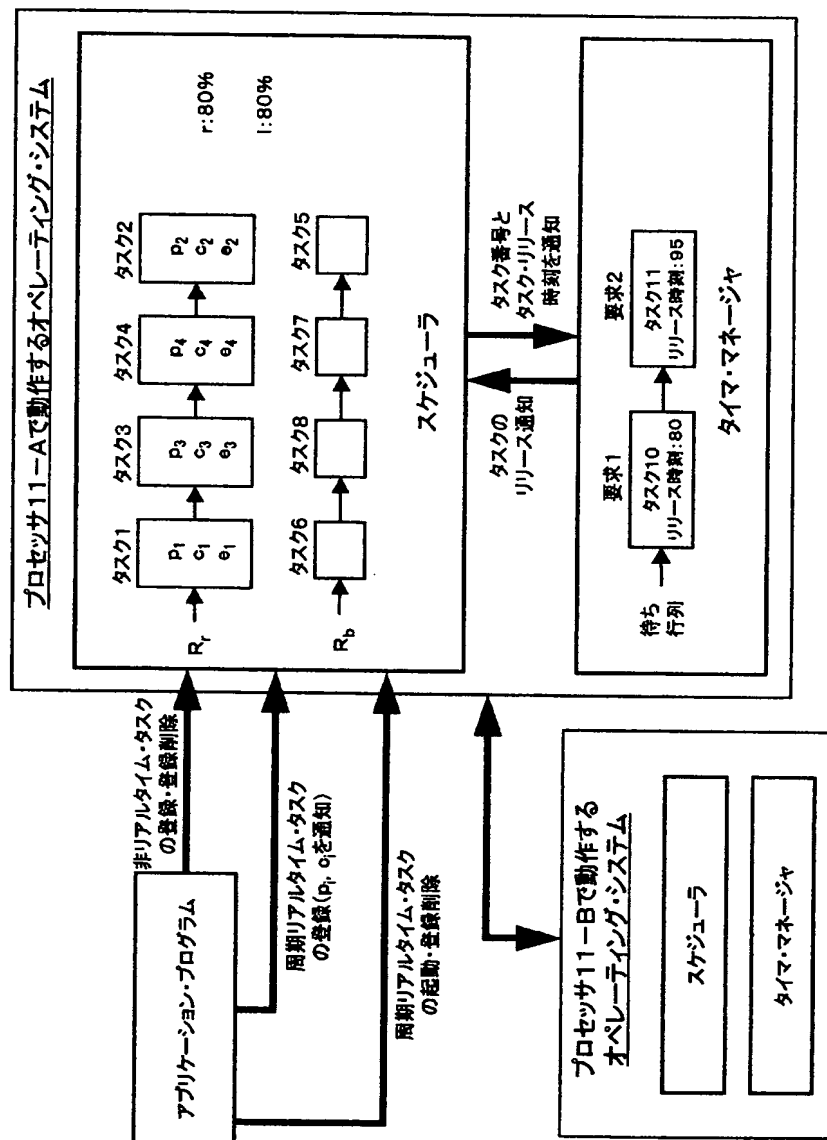
【図4】



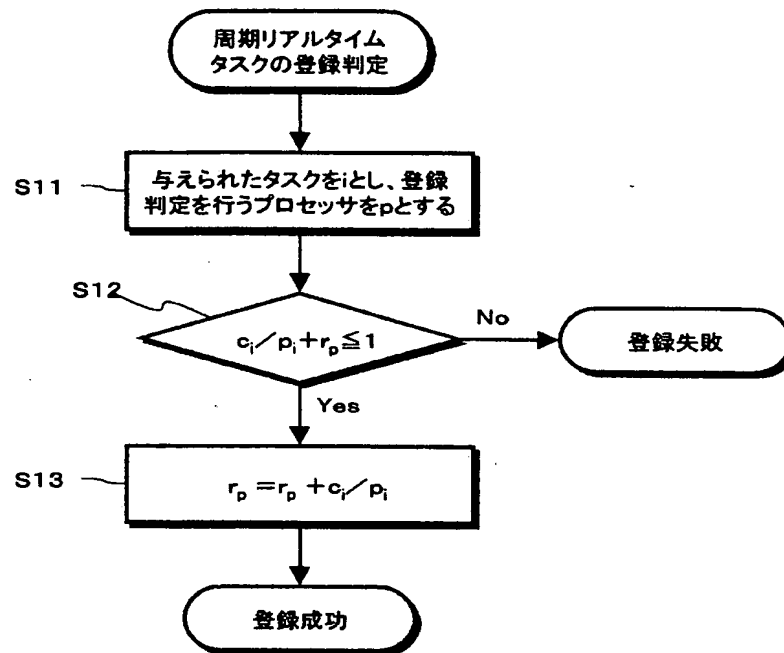
【図8】



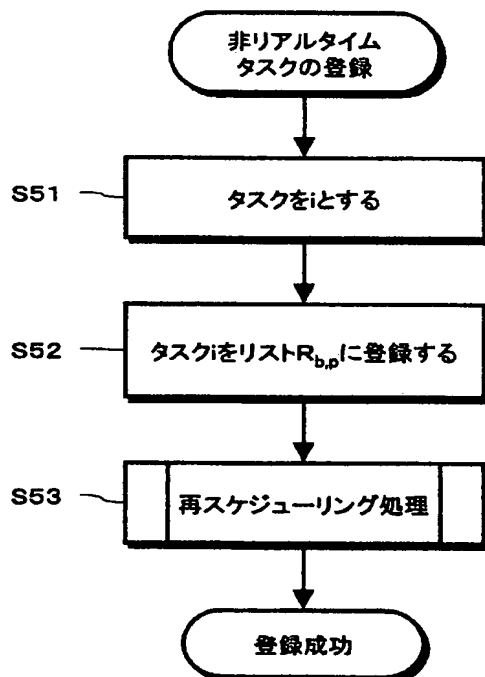
【図3】



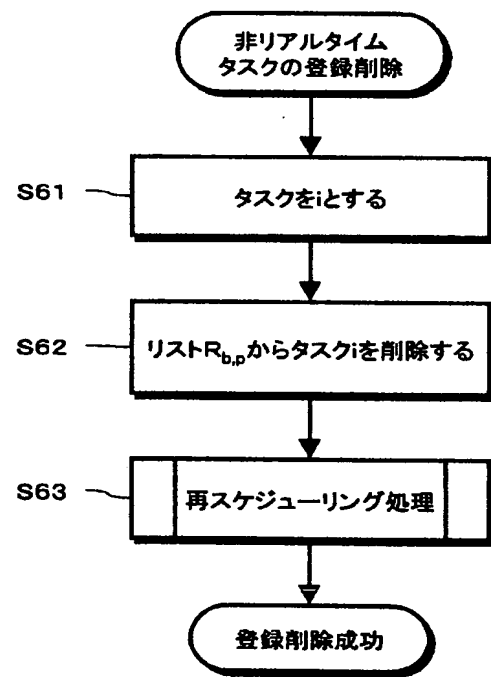
【図5】



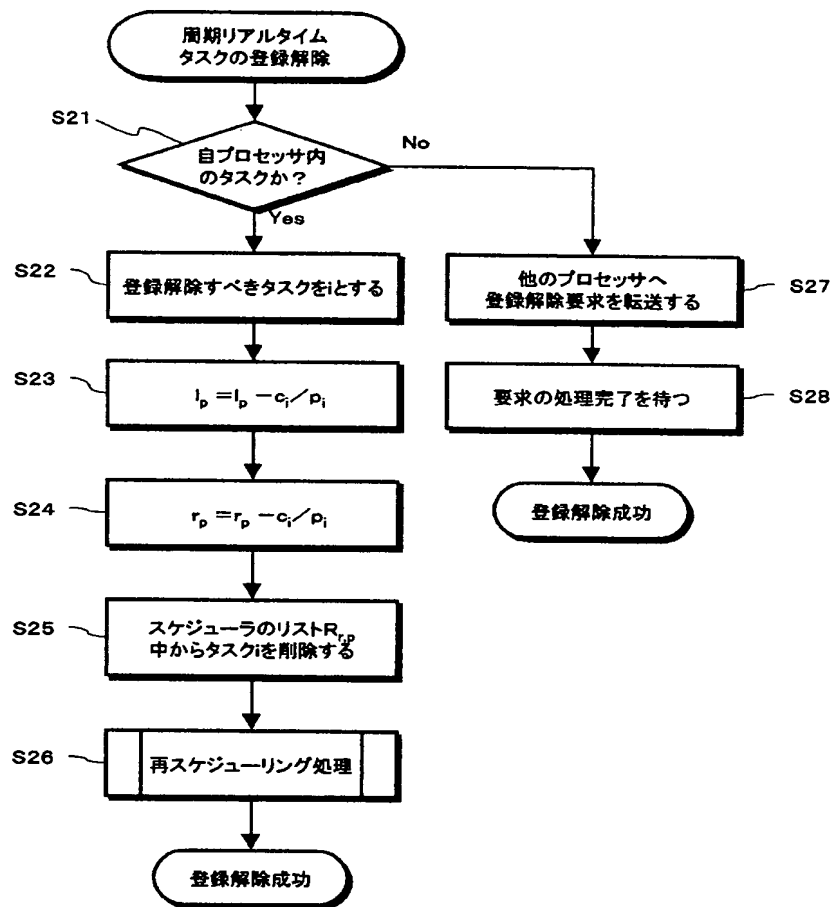
【図9】



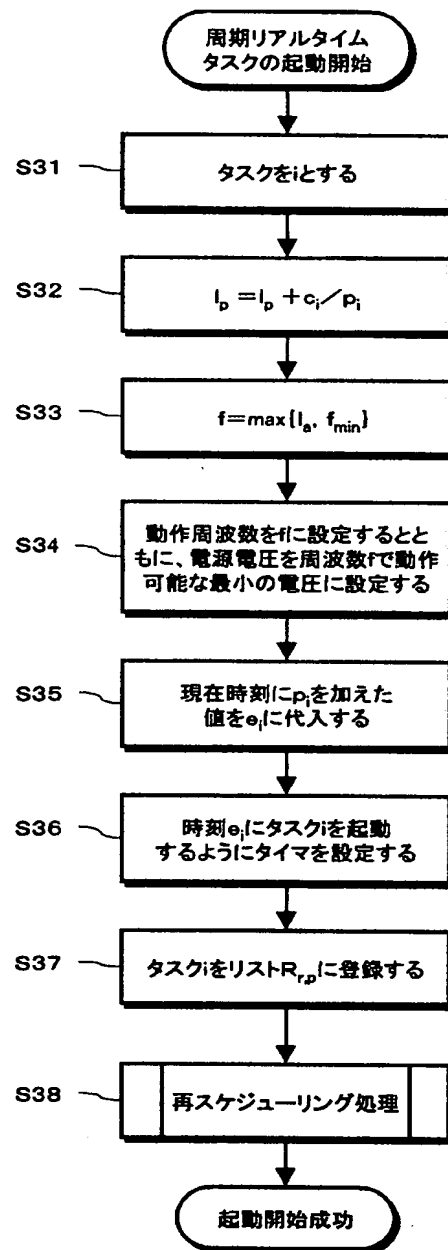
【図10】



【図 6】



【図 7】



【図11】

